```
RRRRRRRRRRR      UUU          UUU  NNN          NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN          NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN          NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRR          RRR UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNNNNN       NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNNNNN       NNN  000         000  FFF              FFF
RRR          RRR UUU          UUU  NNNNNN       NNN  000         000  FFF              FFF
RRRRRRRRRRRR     UUU          UUU  NNN  NNN     NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN  NNN     NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN  NNN     NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRR    RRR       UUU          UUU  NNN       NNNNNN  000         000  FFF              FFF
RRR    RRR       UUU          UUU  NNN       NNNNNN  000         000  FFF              FFF
RRR    RRR       UUU          UUU  NNN       NNNNNN  000         000  FFF              FFF
RRR       RRR    UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR       RRR    UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR       RRR    UUU          UUU  NNN          NNN  000         000  FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
RRR          RRR UUUUUUUUUUUUUUU  NNN          NNN     000000000     FFF              FFF
```

```
_$2

Sym
---
NDX
NDX
NUM
NUM
OPE
OUT
PAC
PAC
PAC
PAC
PAC
PAC
PAC
PAD
PAG
PAG
PAG
PAG
PAG
PAG
PAG
PER
PUT
RCO
RIN

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR
STR

STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

```
MM      MM    AAAAAA    KK      KK  NN      NN  DDDDDDDD    XX      XX
MM      MM    AAAAAA    KK      KK  NN      NN  DDDDDDDD    XX      XX
MMMM  MMMM  AA      AA  KK      KK  NN      NN  DD      DD  XX      XX
MMMM  MMMM  AA      AA  KK      KK  NN      NN  DD      DD  XX      XX
MM  MM  MM  AA      AA  KK    KK    NNNN    NN  DD      DD    XX  XX
MM  MM  MM  AA      AA  KK    KK    NNNN    NN  DD      DD    XX  XX
MM      MM  AA      AA  KKKKKK      NN  NN  NN  DD      DD      XX
MM      MM  AA      AA  KKKKKK      NN  NN  NN  DD      DD      XX
MM      MM  AAAAAAAAAA  KK    KK    NN    NNNN  DD      DD    XX  XX
MM      MM  AAAAAAAAAA  KK    KK    NN    NNNN  DD      DD    XX  XX
MM      MM  AA      AA  KK      KK  NN      NN  DD      DD  XX      XX     ....
MM      MM  AA      AA  KK      KK  NN      NN  DD      DD  XX      XX     ....
MM      MM  AA      AA  KK      KK  NN      NN  DDDDDDDD    XX      XX     ....
MM      MM  AA      AA  KK      KK  NN      NN  DDDDDDDD    XX      XX     ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 %TITLE 'Processes .INDEX, .ENT, and .SUBI directives.'
;     2          0002  0 MODULE makndx ( IDENT = 'V04-000'
;     3        P 0003  0                 %BLISS32[, ADDRESSING_MODE (EXTERNAL    = LONG_RELATIVE,
;     4          0004  0                                            NONEXTERNAL = LONG_RELATIVE)]
;     5          0005  0                 ) =
;     6          0006  1 BEGIN
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1 !
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:     DSR (Digital Standard RUNOFF) / DSRPLUS
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:     Processes the .INDEX, .ENTRY, and .SUBINDEX commands.
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT:  Transportable
;    37          0037  1 !
;    38          0038  1 ! AUTHOR:       R.W.Friday       CREATION DATE: June, 1978
;    39          0039  1 !
;    40          0040  1
```

```
;    42          0041  1 %SBTTL 'Revision History'
;    43          0042  1 !
;    44          0043  1 ! MODIFIED BY:
;    45          0044  1 !
;    46          0045  1 !       018     REM00018        Ray Marshall    17-November-1983
;    47          0046  1 !               Modified the external definition of ATABLE to use the new
;    48          0047  1 !                 macro ATABLE_DEFINITION defined in ATCODE.REQ.
;    49          0048  1 !
;    50          0049  1 !       017     KAD00017        Keith Dawson    23-March-1983
;    51          0050  1 !               Changed GCA_FLIP bit to (.gca_op_dev EQL op_dev_flip).
;    52          0051  1 !
;    53          0052  1 !       016     KFA00016        Ken Alden       16-March-1983
;    54          0053  1 !               SCA must now be initialized by hand all the time
;    55          0054  1 !                 since DSR is now capable of doing SAVE/RESTORE.
;    56          0055  1 !
;    57          0056  1 !       015     KFA00015        Ken Alden       10-March-1983
;    58          0057  1 !               Fixed bug with .X, CHSSTR should not have been called.
;    59          0058  1 !
;    60          0059  1 !       014     RER00014        Ron Randall     07-March-1983
;    61          0060  1 !               Global edit of all modules. Updated module names, idents,
;    62          0061  1 !               copyright dates. Changed require files to BLISS library.
;    63          0062  1 !
;    64          0063  1 !--
;    65          0064  1
```

```
;    67          0065  1 %SBTTL 'Module Level Declarations'
;    68          0066  1
;    69          0067  1 !
;    70          0068  1 ! TABLE OF CONTENTS:
;    71          0069  1 !
;    72          0070  1 ! INCLUDE FILES:
;    73          0071  1 !
;    74          0072  1 LIBRARY 'NXPORT:XPORT';                 ! XPORT Library
;    75          0073  1 REQUIRE 'REQ:RNODEF';                   ! RUNOFF variant definitions
;    76          0204  1
;    77        U 0205  1 %IF DSRPLUS %THEN
;    78        U 0206  1 LIBRARY 'REQ:DPLLIB';                   ! DSRPLUS BLISS Library
;    79          0207  1 %ELSE
;    80          0208  1 LIBRARY 'REQ:DSRLIB';                   ! DSR BLISS Library
;    81          0209  1 %FI
;    82          0210  1
;    83          0211  1 !
;    84          0212  1 ! EXTERNAL REFERENCES:
;    85          0213  1 !
;    86          0214  1 EXTERNAL LITERAL
;    87          0215  1     RINTES : UNSIGNED (8);
;    88          0216  1
;    89          0217  1 EXTERNAL
;    90          0218  1     atable : atable_definition, ! Action table.  Used to identify what type of
;    91          0219  1                                ! action is to be taken on encountering any
;    92          0220  1                                ! given character.
;    93          0221  1     FLGT : FLAG_TABLE,
;    94          0222  1
;    95        U 0223  1 %IF DSRPLUS %THEN
;    96        U 0224  1     FS01 : FIXED_STRING,
;    97          0225  1 %FI
;    98          0226  1
;    99          0227  1     GCA : GCA_DEFINITION,
;   100          0228  1     IRA : FIXED_STRING,
;   101          0229  1     MRA : REF FIXED_STRING,
;   102          0230  1     SCA : SCA_DEFINITION,
;   103          0231  1     TSF : TSF_DEFINITION,
;   104          0232  1     XMRA : FIXED_STRING,
;   105          0233  1     XTSF : VECTOR;
;   106          0234  1
;   107          0235  1 EXTERNAL ROUTINE
;   108          0236  1     ENDWRD,
;   109          0237  1     OFT,
;   110          0238  1     OUTLIN,
;   111          0239  1     RSKIPS,
;   112          0240  1     SCANT,
;   113          0241  1     SETCAS,
;   114          0242  1     SUBXR;
;   115          0243  1
;   116        U 0244  1 %IF DSRPLUS %THEN
;   117        U 0245  1 EXTERNAL ROUTINE
;   118        U 0246  1     ENDCMT,
;   119        U 0247  1     PUTATT,
;   120        U 0248  1     CHSSTR,
;   121        U 0249  1     WRATTR,
;   122        U 0250  1     XPLUS;
;   123          0251  1 %FI
```

```
;  124          0252  1 !
;  125          0253  1 ! OWN STORAGE:
;  126          0254  1 !
;  127          0255  1 OWN
;  128          0256  1     PP_SCA : $H_R_SCA_BLOCK;    !Used in PUSH_SCA, POP_SCA macros (defined in SCA.REQ).
```

```
;  130          0257  1 GLOBAL ROUTINE MAKNDX (HANDLER_CODE, DO_INDEXING) : NOVALUE =
;  131          0258  1
;  132          0259  1 !++
;  133          0260  1 ! FUNCTIONAL DESCRIPTION:
;  134          0261  1 !
;  135          0262  1 !       See ABSTRACT, above.
;  136          0263  1 !
;  137          0264  1 ! FORMAL PARAMETERS:
;  138          0265  1 !
;  139          0266  1 !       HANDLER_CODE    - The command to be processed.
;  140          0267  1 !       DO_INDEXING     - If FALSE, the scan is done but no index entry is made.
;  141          0268  1 !
;  142          0269  1 ! IMPLICIT INPUTS:      None
;  143          0270  1 !
;  144          0271  1 ! IMPLICIT OUTPUTS:     None
;  145          0272  1 !
;  146          0273  1 ! ROUTINE VALUE:
;  147          0274  1 ! COMPLETION CODES:     None
;  148          0275  1 !
;  149          0276  1 ! SIDE EFFECTS:         None
;  150          0277  1 !--
;  151          0278  1
;  152          0279  2     BEGIN
;  153          0280  2     LOCAL
;  154          0281  2         SCA_PARA_COPY,
;  155          0282  2         SCA_INDENT_COPY,
;  156          0283  2         HOLD_IND_FLAG,
;  157          0284  2         HOLD_IND_EFLAG,
;  158          0285  2         SCA_COPY : SCA_DEFINITION;              !Used to preserve SCA.
;  159          0286  2
;  160          0287  2     LOCAL
;  161          0288  2         MRA_TEMP : VECTOR [4 + CH$ALLOCATION (1000)],   !Temporary MRA area
;  162          0289  2         TSF_TEMP : VECTOR [TSF_SIZE];           !Temporary TSF
;  163          0290  2
;  164          0291  2     LOCAL
;  165          0292  2         MRA_ADDRESS,                            !Preserve interrupted MRA
;  166          0293  2         TSF_ADDRESS;                            !Preserve interrupted TSF
;  167          0294  2
;  168          0295  2     LOCAL
;  169          0296  2         XTN;                                    !Computed transaction number for index entry.
;  170          0297  2
;  171        U 0298  2 %IF DSRPLUS %THEN
;  172        U 0299  2     LOCAL
;  173        U 0300  2         AN_XYPLUS;
;  174        U 0301  2
;  175        U 0302  2     AN_XYPLUS = 0;
;  176          0303  2 %FI
;  177          0304  2
;  178          0305  2     XTN = 0;                                    !Assume it's a .ENTRY command.
;  179          0306  2     !No transaction number is needed for .ENTRY commands, since they
;  180          0307  2     !get no page number in the index.  Also, if indexing commands
;  181          0308  2     !are to be ignored, no transaction number is needed either.
;  182          0309  2
;  183          0310  2     IF  .DO_INDEXING
;  184          0311  3         AND (.HANDLER_CODE NEQ H_ENTRY)
;  185          0312  3
;  186        U 0313  3 %IF DSRPLUS %THEN
```

```
;  187          U 0314  3          AND (.HANDLER_CODE NEQ H_YPLUS)
;  188            0315  3 %FI
;  189            0316  3
;  190            0317  2      THEN
;  191            0318  3          BEGIN
;  192            0319  3          ! Compute the transaction number associated with this entry.
;  193            0320  3
;  194            0321  3          XTN = .GCA_NORMAL_XTN;
;  195            0322  3          GCA_NORMAL_XTN = .GCA_NORMAL_XTN + 1;   !Bump for next index entry
;  196            0323  3
;  197            0324  3          !The page number associated with this index entry is the same
;  198            0325  3          !as that of the word being worked on.
;  199            0326  3
;  200            0327  3          IF   .SCA_WRD_F_XTN EQL 0
;  201            0328  3          THEN
;  202            0329  3              SCA_WRD_F_XTN = .XTN;
;  203            0330  3
;  204            0331  3          IF   .XTN NEQ 0
;  205            0332  3          THEN
;  206            0333  3              SCA_WRD_L_XTN = .XTN;
;  207            0334  3
;  208            0335  2          END;
;  209            0336  2
;  210            0337  2          RSKIPS (IRA);                                  !Skip spaces and tabs preceeding the index item(s)
;  211            0338  2
;  212            0339  2          HOLD_IND_FLAG = .FLGT [IND_FLAG, FLAG_CHARACTER];       !Preserve <INDEX flag> status.
;  213            0340  2          HOLD_IND_EFLAG = .FLGT [IND_FLAG, FLAG_ENABLED];
;  214            0341  2          OFT (H_NO_FLAGS_INDE, IND_FLAG);                !Disable <INDEX flag>
;  215            0342  2
;  216            0343  2          IF   .FLGT [SBX_FLAG, FLAG_ENABLED] AND .SCA_FLAGS
;  217            0344  2          THEN
;  218            0345  2          !Enable <SUBINDEX flag> recognition. The <SUBINDEX flag> temporarily
;  219            0346  2          !occupies areas reserved for the <INDEX flag>.
;  220            0347  3              BEGIN
;  221            0348  3              FLGT [IND_FLAG, FLAG_CHARACTER] = .FLGT [SBX_FLAG, FLAG_CHARACTER];
;  222            0349  3              FLGT [IND_FLAG, FLAG_ENABLED] = TRUE;
;  223            0350  3              ATABLE [.FLGT [SBX_FLAG, FLAG_CHARACTER]] = A_FLAG;
;  224            0351  2              END;
;  225            0352  2
;  226          U 0353  2 %IF DSRPLUS %THEN
;  227          U 0354  2      IF   .HANDLER_CODE EQL H_XPLUS OR .HANDLER_CODE EQL H_YPLUS
;  228          U 0355  2      THEN
;  229          U 0356  2          BEGIN
;  230          U 0357  2          LOCAL
;  231          U 0358  2              STATUS;
;  232          U 0359  2
;  233          U 0360  2          STATUS = XPLUS (.HANDLER_CODE);
;  234          U 0361  2
;  235          U 0362  2          IF   .STATUS
;  236          U 0363  2          THEN
;  237          U 0364  2              BEGIN
;  238          U 0365  2              !
;  239          U 0366  2              !Note the fact that an .XP/.YP was given, so we can generate
;  240          U 0367  2              !a warning message if .DX/.PX is ever used after .XP/.YP.
;  241          U 0368  2              GCA_XP_EVER = TRUE;
;  242          U 0369  2              !
;  243          U 0370  2              !Mark this entry as an .XP/.YP entry. This will allow DOFLG
```

```
;  244          U 0371  2              !to determine if it should generate escape sequence for the
;  245          U 0372  2              !Nopermute flag (as it should if the command is .XP/.YP), or
;  246          U 0373  2              !simply eat it (as it should for .X/.Y).
;  247          U 0374  2              AN_XYPLUS = TRUE;
;  248          U 0375  2              !
;  249          U 0376  2              END
;  250          U 0377  2          ELSE
;  251          U 0378  2              BEGIN
;  252          U 0379  2              !The syntax of user-supplied attributes is illegal (mismatched
;  253          U 0380  2              !parens). In this case we cannot distinguish what is intended
;  254          U 0381  2              !for the index entry, so advance past the rest of the text and
;  255          U 0382  2              !return.
;  256          U 0383  2              ENDCMT ();
;  257          U 0384  2              RETURN;
;  258          U 0385  2              END;
;  259          U 0386  2
;  260          U 0387  2          END;
;  261          U 0388  2
;  262             0389  2 %FI
;  263             0390  2      PUSH_SCA;   !Save the SAVEd SCA bits.
;  264             0391  2
;  265             0392  2      INCR I FROM 0 TO SCA_SIZE - 1 DO
;  266             0393  2          SCA_COPY [.I] = .SCA [.I];                  !Save SCA
;  267             0394  2
;  268             0395  2      !Initialize the SCA before doing SCANT
;  269             0396  2      SCA_LM = 0;
;  270             0397  2      SCA_RM = 150;                                  !Set wide right margin so won't start a new line.
;  271             0398  2      SCA_X_FLAG = FALSE;                            !Turn off index flag now in case we are in the middle of an
;  272             0399  2      SCA_SPACING = 1;                               !Single spacing avoids spurious intermediate code.
;  273             0400  2      SETCAS (.GCA_XCASE);                           !Use case translation rules defined by .XLOWER or .XUPPER co
;  274             0401  2      SCA_FC = TRUE;                                 !This is the first character for the MRA.
;  275             0402  2      SCA_FC_CASE = TRUE;                            !Use first character-of-word case rules.
;  276             0403  2      SCA_XROUTINE = TRUE;                           !TRUE ==> use SUBXR for index phrase processing routine.
;  277             0404  2      SCA_PRESCAN = TRUE;                            !A ':' terminates the scan.
;  278             0405  2      SCA_PARA_COPY = .SCA_PARA_PND;                 !Don't let FCIMRA do any indentation, etc.
;  279             0406  2      SCA_INDENT_COPY = .SCA_INDENT;
;  280             0407  2      SCA_PARA_PND = FALSE;
;  281             0408  2      SCA_INDENT = 0;
;  282             0409  2      SCA_FILL = FALSE;
;  283             0410  2      SCA_JUSTIFY = FALSE;
;  284             0411  2
;  285             0412  2      !Preserve the addresses of the structures to which these items refer.
;  286             0413  2      MRA_ADDRESS = .MRA;
;  287             0414  2      TSF_ADDRESS = .TSF;
;  288             0415  2      !Now switch working buffers, so that text currently being accumulated
;  289             0416  2      !is not disturbed.
;  290             0417  2      MRA = MRA_TEMP;
;  291             0418  2      TSF = TSF_TEMP;
;  292             0419  2      !Now manually set up MRA, to look like a FIXED STRING.
;  293             0420  2      FS_MAXSIZE (MRA) = 1000;
;  294             0421  2      FS_INIT (MRA);                                 !And that finishes the initialization.
;  295             0422  2
;  296             0423  2      !Now initialize TSF for SCANT.
;  297             0424  2      INCR I FROM 0 TO TSF_SIZE - 1 DO
;  298             0425  2          TSF [.I] = 0;
;  299             0426  2
;  300          U 0427  2 %IF DSRPLUS %THEN
```

```
;  301        U 0428  2      TSF_XYPLUS = .AN_XYPLUS;                !This information is used by UNPUS (called from OUTLIN), among other
;  302          0429  2  %FI
;  303          0430  2
;  304          0431  2      TSF_INDEX = TRUE;                           !This TSF record describes an index entry.
;  305          0432  2      TSF_FIRST_XTN = .XTN;
;  306          0433  2
;  307          0434  2      SCA_WRD_INT_L = 0;
;  308          0435  2      SCA_WRD_EXT_L = 0;
;  309          0436  2      SCA_WRD_ISEQN = 0;
;  310          0437  2      SCA_WRD_DRAFT = 0;
;  311          0438  2      SCA_WRD_DRAFT_F = 0;
;  312          0439  2      SCA_WRD_BARS = 0;
;  313          0440  2      SCA_WRD_BAR_CHR = 0;
;  314          0441  2      SCA_WRD_SEQN_F = 0;
;  315          0442  2      SCA_WRD_IPAGEN = 0;
;  316          0443  2      SCA_WRD_FOOTW = 0;
;  317          0444  2      SCA_WRD_F_XTN = 0;
;  318          0445  2      SCA_WRD_L_XTN = 0;
;  319          0446  2      SCA_WRD_LC_PNCT = 0;
;  320          0447  2      SCA_WRD_LST_SP = 0;
;  321          0448  2      SCA_WRD_LST_JUS = 0;
;  322          0449  2      SCA_WRD_LST_UND = 0;
;  323          0450  2      SCA_WRD_PNTR = .FS_NEXT (MRA);
;  324          0451  2      SCA_WRD_CPEND = RINTES;                     !Tell ENDCHR no character is pending.
;  325          0452  2      FS_START (MRA) = .SCA_WRD_PNTR;
;  326          0453  2
;  327          0454  2      !Now let SCANT process the command.
;  328          0455  2      SCANT ();
;  329          0456  2
;  330          0457  2      !Restore <INDEX flag> to former status.
;  331          0458  2      OFT (H_NO_FLAGS_INDE, IND_FLAG);
;  332          0459  2      FLGT [IND_FLAG, FLAG_CHARACTER] = .HOLD_IND_FLAG;
;  333          0460  2      FLGT [IND_FLAG, FLAG_ENABLED] = .HOLD_IND_EFLAG;
;  334          0461  2
;  335          0462  2      IF  .FLGT [IND_FLAG, FLAG_ENABLED]
;  336          0463  2          AND .SCA_FLAGS
;  337          0464  2      THEN
;  338          0465  2          ATABLE [.FLGT [IND_FLAG, FLAG_CHARACTER]] = A_FLAG;
;  339          0466  2
;  340          0467  2      !Return here is with a word.
;  341          0468  2      IF  .DO_INDEXING
;  342          0469  2      THEN
;  343          0470  3          BEGIN
;  344          0471  3
;  345          0472  3          IF  .SCA_WRD_CPEND NEQ RINTES
;  346          0473  3          THEN
;  347          0474  3              ENDWRD (FALSE, FALSE, FALSE);       !End word, but don't write a space too.
;  348          0475  3
;  349        U 0476  3  %IF DSRPLUS %THEN
;  350        U 0477  3          !
;  351        U 0478  3          !Find out if there is a sort string,
;  352        U 0479  3          !and if so, check its validity.
;  353        U 0480  3          !
;  354        U 0481  3          IF  .HANDLER_CODE EQL H_XPLUS OR .HANDLER_CODE EQL H_YPLUS
;  355        U 0482  3          THEN
;  356        U 0483  4              BEGIN
;  357        U 0484  4              CHSSTR ();
```

```
;   358          U 0485  3              !
;   359          U 0486  3              ! The sort string has now been checked and adjusted if necessary.
;   360          U 0487  3              ! Now we write the sort= and Append= strings to the attributes block.
;   361          U 0488  3              !
;   362          U 0489  3              WRATTR ();
;   363          U 0490  3              END;
;   364          U 0491  3          !
;   365          U 0492  3          !Write an attributes block only if:
;   366          U 0493  3          ! 1) Indexing is going to the .BRN file, and
;   367          U 0494  3          ! 2) The indexed text was not null, and
;   368          U 0495  3          ! 3) This was an .XP/.YP.
;   369          U 0496  3          ! 4) Not generating a .BFL (FLIP) file.
;   370          U 0497  3          !
;   371          U 0498  3          IF  .GCA_BIX                !Indexing to a file?
;   372          U 0499  3              AND
;   373          U 0500  3              .TSF_EXT_HL NEQ 0       !Non-null index entry?
;   374          U 0501  3              AND
;   375          U 0502  3              .TSF_XYPLUS             !.XP/.YP?
;   376          U 0503  3              AND
;   377          U 0504  3              NOT (.gca_op_dev EQL op_dev_flip)           !Not generating a .BFL?
;   378          U 0505  3          THEN
;   379          U 0506  3              PUTATT ();              !OK, write out the attributes.
;   380            0507  3      %FI
;   381            0508  3
;   382            0509  3          OUTLIN (FALSE);                     !Don't disturb justification algorithm
;   383            0510  2          END;
;   384            0511  2
;   385            0512  2      INCR I FROM 0 TO SCA_SIZE - 1 DO
;   386            0513  2          SCA [.I] = .SCA_COPY [.I];                  !Restore SCA
;   387            0514  2
;   388            0515  2      POP_SCA;    !Restore the SAVEd SCA bits.
;   389            0516  2
;   390            0517  2      MRA = .MRA_ADDRESS;                         !Restore previous status.
;   391            0518  2      TSF = .TSF_ADDRESS;
;   392            0519  2      SCA_PARA_PND = .SCA_PARA_COPY;
;   393            0520  2      SCA_INDENT = .SCA_INDENT_COPY;
;   394            0521  1      END;                                        !End of MAKNDX


                                                        .TITLE  MAKNDX Processes .INDEX, .ENT, and .SUBI direct
;                                                                       ives.
                                                        .IDENT  \V04-000\

                                                        .PSECT  $OWN$,NOEXE,2

                                                00000 PP_SCA: .BLKB   48

                                                        .EXTRN  RINTES, ATABLE, FLGT
                                                        .EXTRN  GCA, IRA, MRA, SCA
                                                        .EXTRN  TSF, XMRA, XTSF
                                                        .EXTRN  ENDWRD, OFT, OUTLIN
                                                        .EXTRN  RSKIPS, SCANT, SETCAS
                                                        .EXTRN  SUBXR

                                                        .PSECT  $CODE$,NOWRT,2

                                            0FFC 00000  MAKNDX:: .ENTRY  MAKNDX, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-  ; 0257
```

```
                                                
```

```
                                D4   BA   D4 0011E          CLRL    @SCA+100                                        ; 0410
                             55 00000000G    EF   D0 00121          MOVL    MRA, MRA_ADDRESS                        ; 0413
                             54 00000000G    EF   D0 00128          MOVL    TSF, TSF_ADDRESS                        ; 0414
                   00000000G EF      00A0    CE   9E 0012F          MOVAB   MRA_TEMP, MRA                           ; 0417
                   00000000G EF              6E   9E 00138          MOVAB   TSF_TEMP, TSF                           ; 0418
                             50 00000000G    EF   D0 0013F          MOVL    MRA, R0                                 ; 0420
                         08  A0      03E8   8F   3C 00146          MOVZWL  #1000, 8(R0)                            ;
                                       0C   A0   D4 0014C          CLRL    12(R0)                                  ; 0421
                             60        10   A0   9E 0014F          MOVAB   16(R0), (R0)                            ;
                         04  A0              60   D0 00153          MOVL    (R0), 4(R0)                             ;
                             52 00000000G    EF   D0 00157          MOVL    TSF, R2                                 ; 0425
                                           51   D4 0015E          CLRL    I                                       ;
                                         6241   D4 00160 5$:      CLRL    (R2)[I]                                 ;
                      F9     51              27   F3 00163          AOBLEQ  #39, I, 5$                              ;
                         14  A2              01   D0 00167          MOVL    #1, 20(R2)                              ; 0431
                         38  A2              53   D0 0016B          MOVL    XTN, 56(R2)                             ; 0432
                                     6C      AA   7C 0016F          CLRQ    SCA+252                                 ; 0434
                                     74      AA   7C 00172          CLRQ    SCA+260                                 ; 0436
                                     7C      AA   D4 00175          CLRL    SCA+268                                 ; 0438
                               0080  CA            01   8A 00178          BICB2   #1, SCA+272                     ; 0439
                                     0084    CA   D4 0017D          CLRL    SCA+276                                 ; 0440
                                     008C    CA   7C 00181          CLRQ    SCA+284                                 ; 0441
                                     0094    CA   7C 00185          CLRQ    SCA+292                                 ; 0443
                                     009C    CA   D4 00189          CLRL    SCA+300                                 ; 0445
                                     00B8    CA   7C 0018D          CLRQ    SCA+328                                 ; 0446
                                     00C0    CA   7C 00191          CLRQ    SCA+336                                 ; 0448
                               68    AA        04   A0   D0 00195          MOVL    4(R0), SCA+248                  ; 0450
                             0088    CA        00G  8F   9A 0019A          MOVZBL  #RINTES, SCA+280                ; 0451
                             60              68   AA   D0 001A0          MOVL    SCA+248, (R0)                   ; 0452
                   00000000G EF                   00   FB 001A4          CALLS   #0, SCANT                       ; 0455
                                                  0A   DD 001AB          PUSHL   #10                             ; 0458
                                        7E        84   8F   9A 001AD          MOVZBL  #132, -(SP)                     ;
                   00000000G EF                   02   FB 001B1          CALLS   #2, OFT                         ;
                   00000000G EF                   57   D0 001B8          MOVL    HOLD_IND_FLAG, FLGT+112         ; 0459
                   00000000G EF                   56   D0 001BF          MOVL    HOLD_IND_EFLAG, FLGT+40         ; 0460
                             13 00000000G    EF   E9 001C6          BLBC    FLGT+40, 6$                             ; 0462
                             0F        00    BA   E9 001CD          BLBC    @SCA+144, 6$                            ; 0463
                             50 00000000G    EF   9E 001D1          MOVAB   ATABLE, R0                              ; 0465
                   00000000GFF40             03   90 001D8          MOVB    #3, @FLGT+112[R0]                       ;
                             1F        08    AC   E9 001E0 6$:      BLBC    DO_INDEXING, 8$                         ; 0468
                   00000000G 8F      0088    CA   D1 001E4          CMPL    SCA+280, #RINTES                        ; 0472
                                             0B   13 001ED          BEQL    7$                                      ;
                                           7E   7C 001EF          CLRQ    -(SP)                                   ; 0474
                                           7E   D4 001F1          CLRL    -(SP)                                   ;
                   00000000G EF                   03   FB 001F3          CALLS   #3, ENDWRD                      ;
                                           7E   D4 001FA 7$:      CLRL    -(SP)                                   ; 0509
                   00000000G EF                   01   FB 001FC          CALLS   #1, OUTLIN                      ;
                                           50   D4 00203 8$:      CLRL    I                                       ; 0512
                         FF70 CA40       FE80 CD40   D0 00205 9$:      MOVL    SCA_COPY[I], SCA[I]                     ; 0513
                      EF                 50 0000005F    8F   F3 0020E          AOBLEQ  #95, I, 9$                      ;
                                   D4    BA              6B   D0 00216          MOVL    PP_SCA, @SCA+100                ;
                                   D8    BA        04    AB   D0 0021A          MOVL    PP_SCA+4, @SCA+104              ;
                                   DC    BA        08    AB   D0 0021F          MOVL    PP_SCA+8, @SCA+108              ;
                                   E0    BA        0C    AB   D0 00224          MOVL    PP_SCA+12, @SCA+112             ;
                                   E4    BA        10    AB   D0 00229          MOVL    PP_SCA+16, @SCA+116             ;
                                   E8    BA        14    AB   D0 0022E          MOVL    PP_SCA+20, @SCA+120             ;
                                   EC    BA        18    AB   D0 00233          MOVL    PP_SCA+24, @SCA+124             ;
```

```
                     F0   BA        1C   AB  D0 00238         MOVL    PP_SCA+28, @SCA+128                    ;
                     F4   BA        20   AB  D0 0023D         MOVL    PP_SCA+32, @SCA+132                    ;
                     F8   BA        24   AB  D0 00242         MOVL    PP_SCA+36, @SCA+136                    ;
                     FC   BA        28   AB  D0 00247         MOVL    PP_SCA+40, @SCA+140                    ;
                     00   BA        2C   AB  D0 0024C         MOVL    PP_SCA+44, @SCA+144                    ;
            00000000G   EF                 55  D0 00251         MOVL    MRA_ADDRESS, MRA                       ; 0517
            00000000G   EF                 54  D0 00258         MOVL    TSF_ADDRESS, TSF                       ; 0518
                     4C   AA                 58  7D 0025F         MOVQ    SCA_INDENT_COPY, SCA+220               ; 0520
                                                 04 00263         RET                                            ; 0521

; Routine Size:  612 bytes,    Routine Base:  $CODE$ + 0000

;  395          0522  1
;  396          0523  1 END                                     !End of module
;  397          0524  0 ELUDOM



;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $OWN$                          48  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        612  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                               Library Statistics
;
;                                  -------- Symbols --------      Pages       Processing
;       File                       Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]XPORT.L32;1     590         0         0         252       00:00.1
;  _$255$DUA28:[RUNOFF.SRC]DSRLIB.L32;1    1248        87         6          86       00:00.3



;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:MAKNDX/OBJ=OBJ$:MAKNDX MSRC$:MAKNDX/UPDATE=(ENH$:MAKNDX)

; Size:         612 code + 48 data bytes
; Run Time:         00:16.4
; Elapsed Time:     00:36.0
; Lines/CPU Min:     1920
; Lexemes/CPU-Min: 18557
; Memory Used:  144 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY